NEW SCIENTIST

DIY software

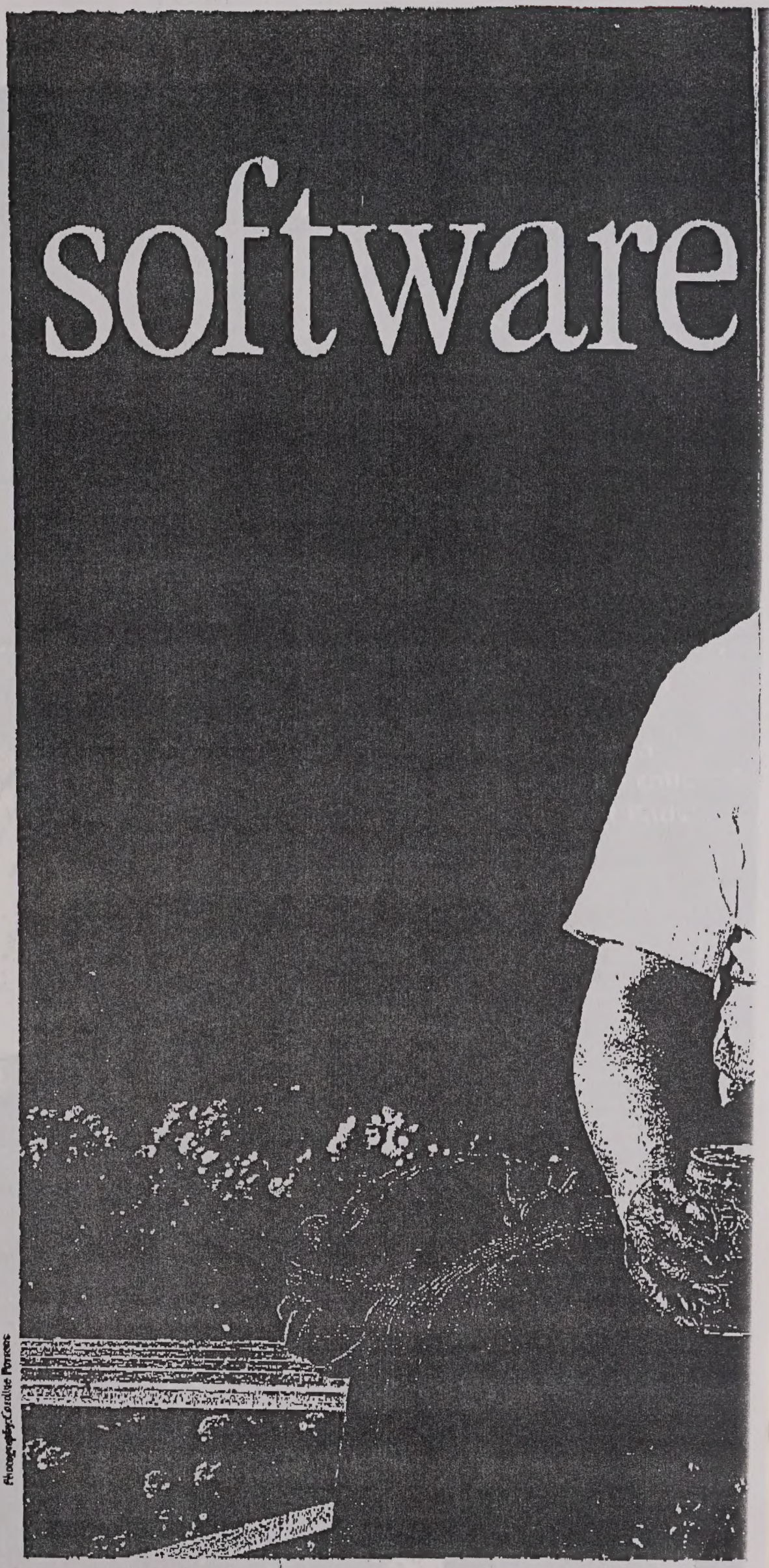
Imagine programs so simple a child can build them. It may sound like a dream, but *Bob Johnstone* meets the Japanese professor who is hard at work trying to make it come true

Hokkaido is an unlikely place from which to start a revolution. Japan's nothernmost island is better known for winter sports and ice sculptures than high tech. But the aim of Yuzuru Tanaka, a 45-year-old professor of electrical engineering at Hokkaido University, is to revolutionise the way software is developed through a concept he calls Intelligent Pads.

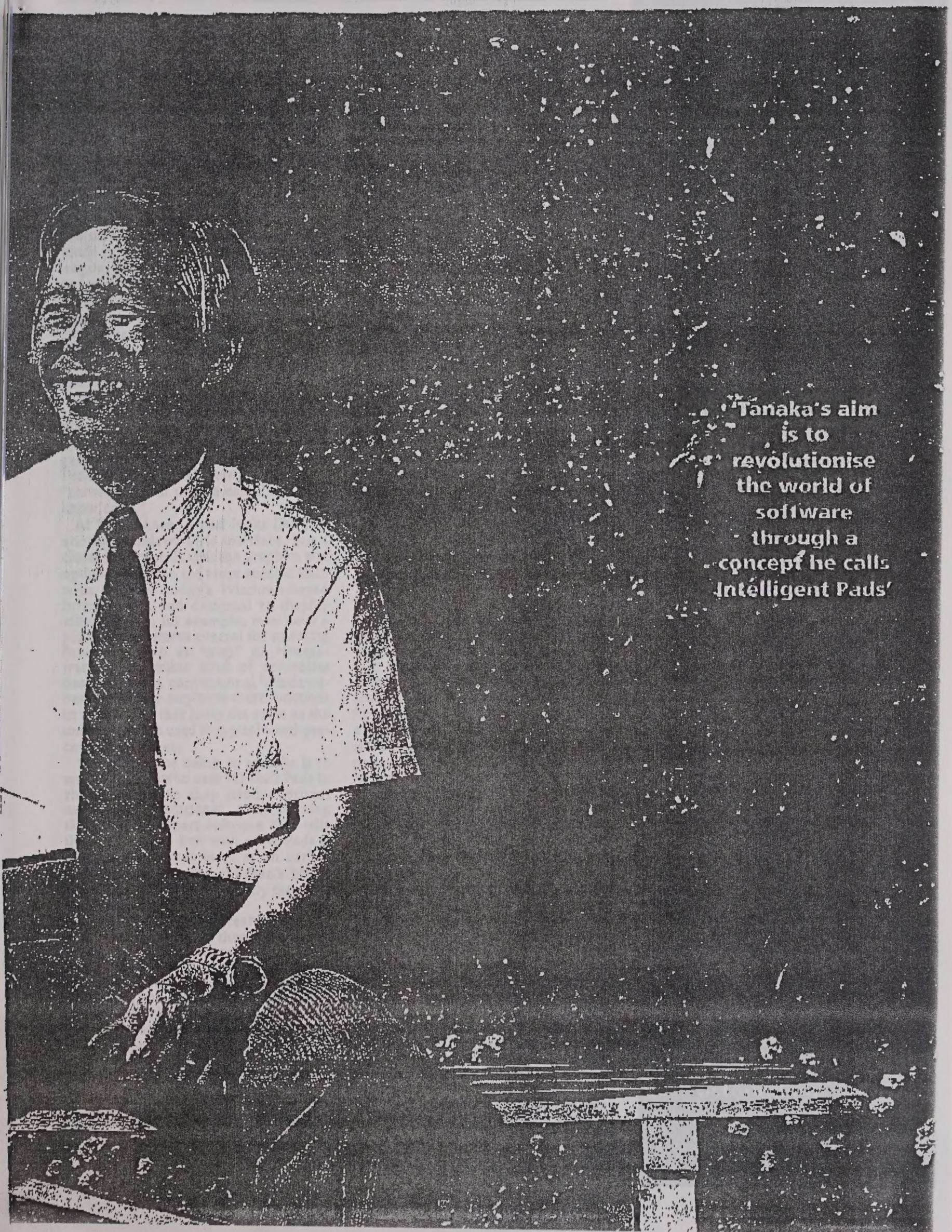
This is ambition on a grand scale. First, Tanaka is an academic, not an entrepreneur, secondly, he is a Japanese academic burdened by the stereotype that Japanese can't write software, and thirdly, he is an academic at a provincial university, hundreds of kilometres north of Japan's commercial and administrative heart. Ultimately, if Intelligent Pads are to succeed, then they must find a niche in an environment that is dominated by American giants such as Microsoft and Apple.

What are Intelligent Pads, and how do they differ from the offerings of the Americans? The most important difference is that the target of the Microsoft and Apple products is professional software developers, whereas Intelligent Pads are primarily intended for software users. Pads are basically the software equivalent of children's building blocks that can connect in any number of ways to create a variety of different structures with different functions. Tanaka uses the generic term Padware to describe both these "blocks", or small programs, and the "constructions", or applications, in which they are used

Over time, as the number of available blocks—the Pad pool—increases and Intelligent Pads are distributed via the Net, Tanaka envisions that padware will



Photography: Creative Pictures


'Tanaka's aim is to revolutionise the world of software through a concept he calls Intelligent Pads'

mutate and evolve in classic Darwinian fashion. Users will decompose the Pads they download, pasting on other Pads, enriching their content. Suppose, for example, a botanist creates a pad on which he pastes a picture of a flower, together with data about the flower's habitat and nutritional requirements. He sends this pad to an entomologist colleague, who adds a transparent video pad of an insect, and uses the resulting combination to personalise his e-mail. Among the recipients are several people who alter the Pads for their own purposes and pass them along, too. Some will do this for fun, others will publish Pads for money.

"Pads are meme media," Tanaka claims, borrowing evolutionary biologist Richard Dawkins' coinage for a unit of cultural transmission equivalent to the gene. "My dream is to make Intelligent Pads a kind of social interface," he says, "through which people can share their knowledge."

At first glance, a pad looks like just another Windows-style interface. But a closer look reveals that the graphics and options available are more specific than with most of today's Windows-based interfaces. A pad designed to display video images, for example, may have a picture of a remote control for selecting functions such as "play" or "freeze-frame". A similar kind of controller designed for a conventional Windows-based interface may have these functions in a menu bar that looks the same as the menu selection used in typical word-processing packages.

The reason for being so specific is to enable anyone who uses Tanaka's Pads to change them as they need—without using any traditional programming skills. All the smart software that tells the Pads how to link with other Pads is written by programmers at an earlier stage, rather like the children's bricks being designed with exactly the right connectors to fit together.

Another interesting feature of the Pads, is that several of them may be run at one time and interact with each other while they are running. If, for example, you have a video of a ball falling on the video pad, you could overlay this with a transparent plotter pad that tracks the ball's movements—so coloured dots would appear on top of the video picture. These movements can then be fed into a third pad on which a graph of the ball's trajectory is being drawn.

Applications like this make perfect tools for teachers. In fact, the video of the ball falling and the software for trajectory prediction have already been developed, by Takafumi Noguchi, an assistant professor of electrical engineering at Kushiro Technical College in Hokkaido, for teaching students basic dynamics.

Out in the business world, companies of all sorts could use Pads to manage their documentation as it flows from desk to desk. And software vendors may find Pads a useful extra for their existing programs if they want to offer very specific functions—such as a video-playing option.

Over time, anyone can build up their own pool of Pads, either on their home or office computer. On a more global scale, since there are no copyright restrictions, Pads on a server for an internal or international network could attract people in far-flung places to build on your work. Tanaka envisions Pads being swapped over the Internet, much as children trade plastic toys from cereal packets in the school playground. Only in this instance, the "toys" can be modified before they are passed on.

Child's play

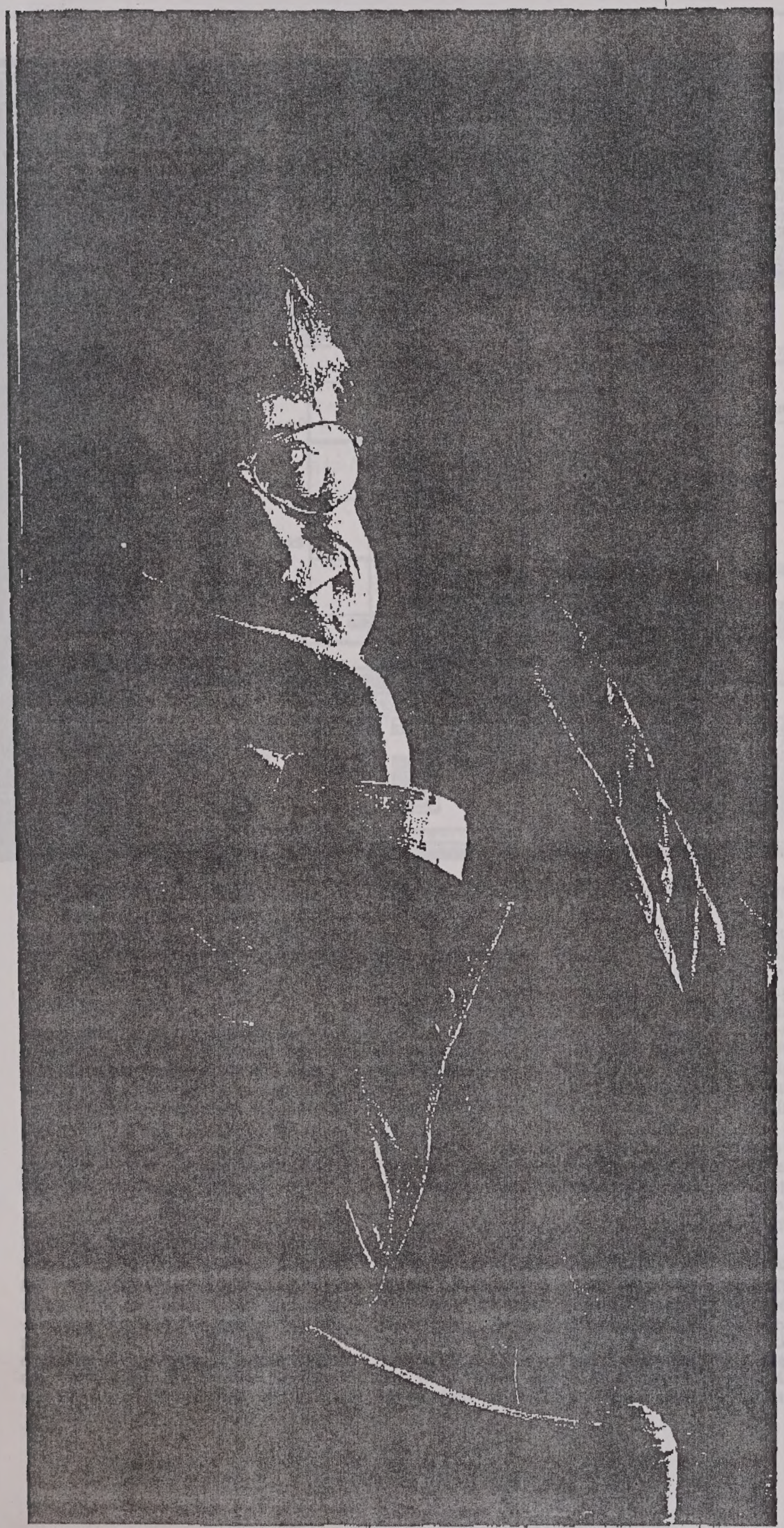
That image could well be the key to success for the Pads: they have a superficially childlike simplicity about them which could entice users in droves.

In other ways, Pads don't differ in broad concept from Microsoft's Object Linking and Embedding (OLE) and Apple's OpenDoc. All three are part of a movement away from "giant software". And the core to all three is "components"—or "parts" in OpenDoc, "containers" in OLE, and, of course, "Pads" in Intelligent Pads. Unlike the giant software, where a single program can take up some 15 megabytes of memory, none of these "chunks" of code takes up more than a couple of megabytes of memory.

The old-style giant software was the result of traditional software programming methodologies, which were mainly procedural: the programmer started at the top and then added "ingredients" as required—rather like following a recipe—in an attempt to take account of all eventualities in a single program.

The component approach of OLE, OpenDoc and Intelligent Pad has a number of advantages over procedural programming, advantages encompassed in a style of programming known as object-oriented programming. This approach allows the programmer some freedom from trying to think about all

'The potential of Pads for networking people across the world is astounding'



possible connections and outcomes before creating a program by writing some standard connection software and then plugging in the application components as necessary.

Changing a component is also easy—you just bolt in a new one to the underlying program. You don't need to think how these extra few lines of code will affect all the lines above and below the change. Because these components carry out specific tasks, ("run video", "track moving object"), you only need the chunks of code that are relevant to the tasks you want to perform.

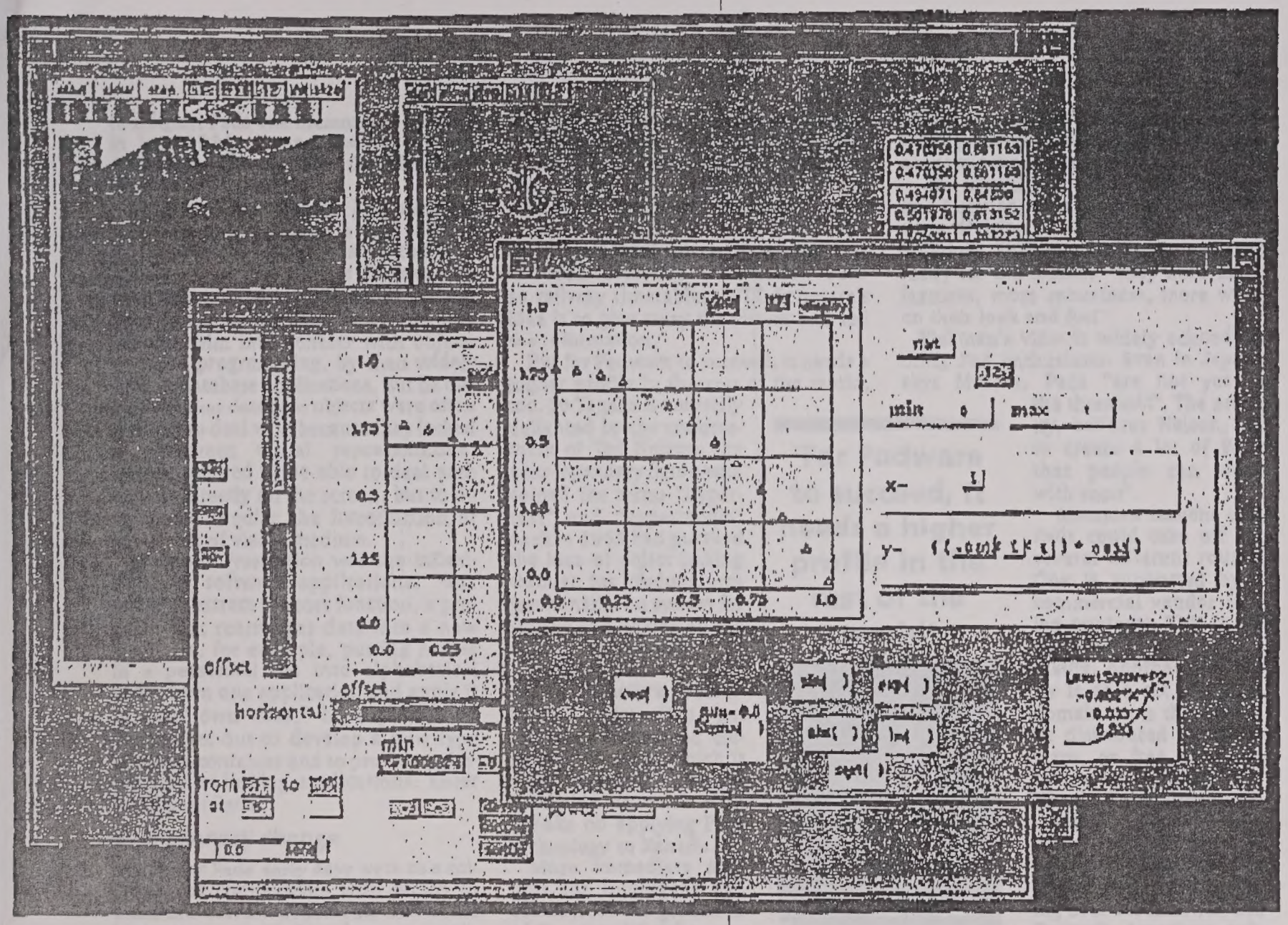
Of the two offerings from the American giants, according to industry analysts, OLE is the least componentised. OLE is aimed mainly at people using the Microsoft Windows environment. It is also designed for software developers creating applications rather than for users. Underlying this application software there is a framework into which the components can be plugged. At the same time, the framework offers links to the standard Microsoft Windows and often network connections as well.

OpenDoc is similar to OLE in many ways. But the OpenDoc consortium which includes Apple, IBM and Novell claims that it will include all the OLE functions as well as some extra features including links into software being developed by Taligent, the Apple and IBM spin-off set up to design new software environments. Apple also emphasises that OpenDoc's components are simpler than OLE's so the developer has more freedom to customise the software.

When it comes to marketing this component approach to software, however, Microsoft leads the field because its Windows is the dominant presentation medium for desktop computing.

So where does that leave Tanaka's padware? Tanaka already has some strong backing in the shape of two of Japan's top computer companies—Hitachi and Fujitsu—thanks to some of his former students spreading the word. Indeed, the story of padware has spread beyond Japan, and developers at these companies' subsidiaries in Britain and the US are testing out the approach.

Tanaka also has the support of the Intelligent Pad Consortium, a nonprofit, nongovernment-sponsored, grass-roots organisation with more than 50 corporate and individual members. And last but not least, Ted Nelson—the



charismatic originator of the concept of distributed hypermedia, and still a champion of involving users in designing their own software—recently moved to Hokkaido to work with Tanaka.

Beauty contest

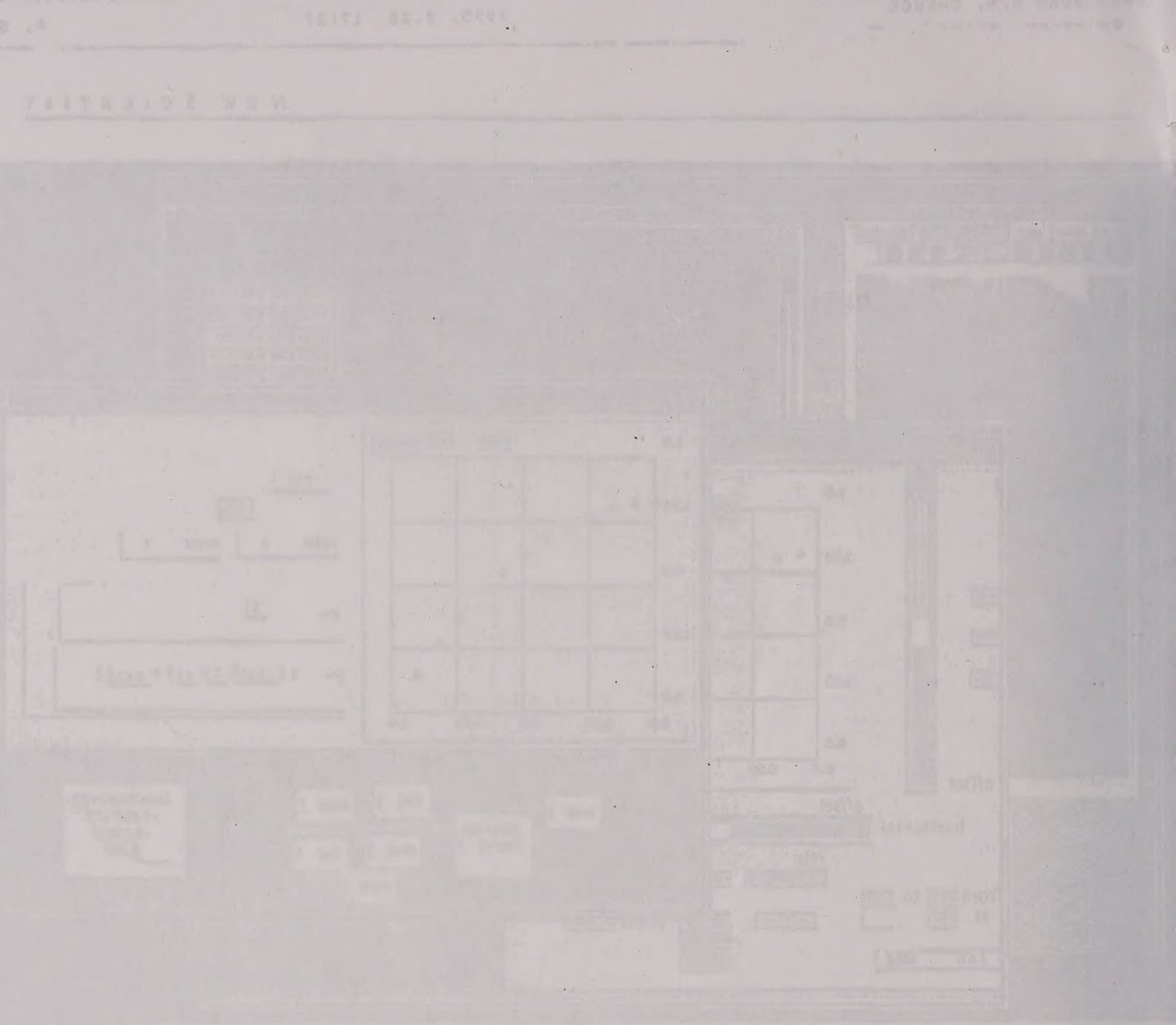
The burning question now as the Pads turn into a product is how can they differentiate themselves from the likes of OLE to gain a share of the market before one of the others becomes a standard by default? A lot will depend on the quality of the software, and the range of applications that users develop. Back to the technical beauty competition.

Supporters of Tanaka believe that the Pads approach has the upper hand in two key areas. Pads take the idea of componentware to its limits. They are designed for developing discrete bits of software using a mouse, some icons and commands written in the user's own language—English, Spanish, Kanji or whatever. And giving users the tools to create their own components puts Tanaka a step ahead of Microsoft and Apple.

But when it comes to technical advantages versus marketing, Dave Simmons, the chief executive of Quasar Knowledge Systems based near Washington DC, is cautious. His company developed a version of Smalltalk, the first programming language designed for building with objects, in which many of the most advanced Intelligent Pads applications are written. "For a while Intelligent Pads was leaps and bounds ahead of anything in the US," he says, "two years ago, they were leading the way." But, he adds, for a variety of reasons, including the Japanese-can't-write-software stereotype and the not-invented-here syndrome, software developers in the US were reluctant to back Intelligent Pads, allowing Microsoft et al time to catch up.

But Simmons has confidence in Tanaka's Pads finding a niche especially because of the three, Pads offer naive users with no programming experience the easiest to understand, ready-made tools with which they can build what they want. There are also plenty of applications on their way. Some come from major corporations such as the Japanese drugs company which is developing a suite of Pads to help it create all the detailed documentation required by law each time a new drug is developed or tested. There are also a number of independent software vendors, who will develop packaged Intelligent Pads applications. But perhaps most important of all, there are the applications written by ordinary users, including individuals.

It is this group rather than the specialists who will benefit most from Pads, believes Hokkaido-based software developer Toshifumi Murata, who is also a member of the Intelligent Pads Consortium's steering committee. The reason is that Padware is accessible to amateurs because it can be built visually, on-screen, without having to write any code. This has long been the dream of many software developers including Bill Atkinson, who created HyperCard for the Macintosh while working for Apple in the early 1980s. Coincidentally, Tanaka says he had the original idea for



Intelligent Pads one memorable evening in late 1986, around the time when Atkinson was putting the finishing touches to HyperCard.

At that time Tanaka had recently returned to Japan after a year visiting IBM's Yorktown Heights laboratories in upstate New York. There Tanaka had been able to expand his horizons in database research. His interest in this area brought him into contact with object-oriented programming, by then widely used in database applications. But he discovered that database objects were often difficult to deal with because they lacked a persistent visual representation. Tanaka wanted to be able to deal with objects directly on the screen, but to do so would require the introduction of some sort of visual medium.

A second frustration was the inflexibility of software applications. "You couldn't extract the sort function, a program that rearranges data into a new sequence, for example, putting names in a personnel file into alphabetical order from one application and apply it to your own data," he explains. So Tanaka set out to develop a new type of media container and to provide a simple way of linking applications. Enter Intelligent Pads.

Evolutionary change

Intelligent Pads' early days were as a collection of components with which to build multimedia documents that could incorporate sound, video footage, graphics and animation. Initially, Tanaka expected that the number of components required would saturate at a certain number. This turned out not to be true. Users—then mostly students in Tanaka's lab—wanted no limits on the way Pads could be designed.

New Pads began to pile up. When they reached around 150, Tanaka noticed a qualitative change. These new Pads were more interesting because they were produced voluntarily rather than as assigned research, and they stimulated other students to develop interesting Pads of their own.

It was at this point, in around 1993, that Tanaka began thinking of Pads development in terms of evolutionary biology, that Pads could evolve rather like organisms. At the same time, he also realised the importance of network distribution as a way of expanding the pool.

And earlier this year Tanaka was able to take stock of progress at a workshop organised by the Intelligent Pads Consortium in Tokyo. Of some 20 demonstrations on display, one was particularly impressive. One was an information viewer put together by NTT Software, the programming arm of Japan's main telecommunications provider, using standard Pads. The viewer enabled users to browse through a database that included everyday information such as railway timetables. NTT Software's idea is to give away the viewer and sell the information.

But for Padware to succeed, it needs a higher profile in the rest of the world, too. So Tanaka is naturally delighted by the endorsement of Ted Nelson, the near-legendary figure who coined the terms "hypertext" and "hypermedia" back in 1965 had pursued the idea of object linking through his life-long project Xanadu—a system for storing and linking all the world's information.

Nelson likes the conceptual structure of Intelligent Pads: "It's got richness, clarity and elegance," he says. Which is what brought him to Hokkaido to work with Tanaka on applying Pads technology to Xanadu.

More immediate support for Intelligent Pad comes from two graduates of Tanaka's lab who now work for Hitachi Software Engineering and Fujitsu. Both have persuaded their managements to commit serious resources to padware development. In Japan, Hitachi has an Intelligent Pads team consisting of 25 programmers and 12 marketing staff. Fujitsu has assigned an equivalent number of people, plus a team of six at an American subsidiary, Fujitsu Open System Solutions, based in San Jose, and another group based at ICL, Fujitsu's British arm, at Bracknell.

As yet, neither group has sold Intelligent Pads software. In Britain, Padware is about to begin a second round of "beta" testing (in which software is distributed to major users free of charge in return for their evaluation before it becomes a product). While in the US, Fujitsu is working with a developer called Speakeasy, which makes a popular mathematical software package that is widely used for economic modelling.

But despite such real-world examples, Dan Fishman, who heads the object software division of Fujitsu Open System Solutions, believes that "Pads are not yet ready for the market—they need more features, more robustness, more work on their look and feel".

Fishman's view is widely echoed by other Pad enthusiasts. Even in Japan, says Murata, Pads "are not yet at the threshold". The priority, believes Nelson, "is to create a lot of Pads that people can work with soon".

'For Padware to succeed, it needs a higher profile in the rest of the world'

Simmons reckons that Pads could take off via several different routes. One is supported by a commercial vendor selling products. This is the approach that Fujitsu is taking. Another possibility is as a set of public domain tools that would be distributed as shareware, or free software. In Japan, Fujitsu and Hitachi have already developed the kernel software—plus 40 types of basic pad—available free of charge for downloading over the Internet, and Fujitsu's American subsidiary is considering doing the same.

One way or another, the Internet could be crucial if Intelligent Pads is to achieve critical mass. "The potential [of Pads] for networking people across the world is astounding," says Nelson. And companies are also getting in on the act. One Japanese company is already planning to distribute Pads featuring animated cartoon characters from "Bit the Cupid", a popular TV series. The Pads will feature puzzles and screen savers.

Such apparently mundane applications may be a long way from Tanaka's vision of reusable intellectual resources. But as any biologist will tell you, sophisticated functions often evolve from the most humble of origins. □

Bob Johnstone covers science and technology in Asia.